

Ai-Based Prediction System for PCOS and Thyroid Disorders in Women using Machine Learning

Amrutha D R – 1SV23CS004, Kavyashree M – 1SV23CS040, Kushala S – 1SV23CS044, Meghanasmitha L – 1SV23CS060

Department of Computer Science and Engineering
Shridevi Institute of Engineering and Technology, Tumakuru, Karnataka, India
Guide: Prof. Mamatha L, Assistant Professor, Department of CSE

Abstract— Polycystic Ovary Syndrome (PCOS) and thyroid disorders are two of the most commonly discussed endocrine conditions in women, and in practice their symptoms often overlap, which makes an early, informal assessment genuinely difficult. In this project we built and actually ran a machine-learning-based screening tool that looks at a patient's profile along with PCOS-related and thyroid-related readings, and returns a preliminary, model-based assessment through a Gradio web interface. Decision Tree and Random Forest were used as the two classification approaches, chosen because one gives us an easy-to-read baseline and the other gives a sturdier ensemble alternative for structured, tabular data of this kind. Beyond the prediction itself, the interface also surfaces nearby specialist suggestions, a chart showing which inputs contributed most to the assessment, and some basic diet/yoga guidance. We include actual screenshots of the running application in this paper — a sample case (a 25-year-old patient from Anekal taluk, Bengaluru Urban) returned a PCOS: NEGATIVE and THYROID: NEGATIVE result, which is reported here purely as a demonstration of the working pipeline and not as a measure of the model's real-world accuracy. Formal accuracy, precision, recall and F1-score figures are left to be filled in once the final experimental run on the complete dataset is available, rather than being guessed at. Overall, the system is meant to work as an academic decision-support prototype — a starting point for further validation — and not as a substitute for laboratory testing or a qualified doctor's opinion.

Index Terms— PCOS, thyroid disorders, machine learning, Decision Tree, Random Forest, Gradio, women's health, disease prediction.

I. INTRODUCTION

A woman's hormonal system touches far more than just reproduction — it affects metabolism, weight, energy levels, mood and menstrual regularity all at once. PCOS is one of the more frequently discussed conditions in this space, usually showing up as some combination of irregular cycles, acne, weight gain and thinning hair [1], while thyroid disorders quietly disturb the body's metabolic rhythm in ways that can look similar on the surface [2]. Because the two conditions share so many overlapping symptoms, it is genuinely easy for either one to be mistaken for, or masked by, the other.

Normally, working out whether someone has PCOS or a thyroid problem means going through a doctor's history-taking, a physical exam, and a set of lab tests — ultrasounds, hormone panels, and so on. Once that information exists in a structured form, though, it becomes something a machine-learning model can actually learn from. That is the basic idea we build on here: instead of a human manually weighing five or six different readings, a trained classifier can weigh them together and return a quick, preliminary read.

What we built is an end-to-end tool rather than just a script that spits out one number. It takes a patient's basic profile, a set of PCOS-related fields, and a set of thyroid-related fields through a Gradio form [3], runs them through the same preprocessing used while training, and returns a report that includes the prediction, nearby specialist suggestions, and some lifestyle pointers. We treat this very much as a student project — a working proof of concept — rather than a finished clinical product.

To be clear from the outset: nothing this tool outputs is a diagnosis. A negative result does not rule a condition out, and a positive one is not a confirmation — both should simply

prompt a proper conversation with a doctor.

II. PROBLEM STATEMENT

The trouble with PCOS and thyroid conditions, from a software point of view, is that neither one comes down to a single symptom or a single number. Trying to hard-code a set of if-else rules for something this multi-factorial gets unwieldy fast, and it doesn't generalize well from one patient to the next. What we needed instead was a pipeline that could take a fairly diverse set of health inputs, put them in a consistent numeric form, learn from past labeled cases, and then produce a sensible prediction for someone new.

On top of that, the same preprocessing used at training time has to be applied, unchanged, whenever a new user fills out the form — otherwise the model would essentially be seeing different kinds of data at prediction time than it learned on. And the whole thing needed to sit behind an interface simple enough that someone with no coding background could use it directly.

III. OBJECTIVES

- Look at which health attributes are actually useful for predicting PCOS and thyroid conditions.
- Clean and prepare a dataset that a supervised classifier can learn from.
- Settle on one consistent feature representation that works for both training and live prediction.
- Build and compare Decision Tree and Random Forest classifiers.
- Check the models with more than one metric — accuracy alone doesn't tell the whole story.

- Wrap the trained pipeline in a Gradio interface that a non-technical user can operate.
- Keep the output firmly labeled as a preliminary, non-clinical prediction.

IV. RELATED WORK AND RESEARCH GAP

A fair amount of prior work applies machine learning to individual health-screening problems using demographic, symptom and biochemical data, and tree-based classifiers keep coming up as a popular choice for this kind of structured, tabular problem [4] — likely because they can pick up nonlinear interactions between variables without needing much manual feature engineering. Decision Trees give a fairly transparent decision structure that is easy to walk through by eye [5]; Random Forest trades a bit of that transparency for the stability of combining many trees [6].

What we noticed while researching this project is that most existing prototypes tend to look at PCOS or thyroid prediction as two completely separate problems, and quite a few of them stop at reporting a single accuracy number without showing what the actual user-facing pipeline looks like end to end. It's also common for these systems to come across as more authoritative than they've actually earned, especially when there's been no outside validation. Our contribution here isn't a new algorithm — it's putting PCOS and thyroid screening into one coherent, reproducible workflow, and being upfront about what is and isn't demonstrated by a single sample run.

V. PROPOSED SYSTEM

The system is organized into five stages that flow into each other: collecting the input, preprocessing it, running inference, generating the result, and presenting it back to the user. On the input side, the Gradio form is split into three collapsible sections — Patient Profile, PCOS Data and Thyroid Data — so the person filling it out isn't overwhelmed by one long form.

Whatever gets typed or selected in the form is converted into the same numeric representation the models were trained on. We were careful to keep this conversion logic identical between training and inference, since even a small mismatch here — say, a different encoding for a symptom checkbox — would quietly break the prediction.

Two classifiers sit behind the interface: a Decision Tree, kept mainly for its interpretability, and a Random Forest, used as the ensemble alternative. Which one ultimately gets used for the deployed prediction should come down to actual validation numbers rather than a general assumption that ensembles are always better.

Once a prediction is made, the report that comes back isn't just a bare PCOS/Thyroid label — it also lists nearby specialists (matched loosely by the taluk entered), a small chart of which inputs contributed most to the result, and some general diet and yoga pointers. These extras make the tool feel more complete, but it's worth being upfront that they are convenience features, not clinically validated recommendations.

VI. SYSTEM ARCHITECTURE

Structurally, the pipeline has seven pieces: data collection, cleaning/preprocessing, feature preparation, model training, model evaluation, the prediction engine itself, and the Gradio

interface on top. During development, the historical dataset moves through the first three stages before the classifiers are fit on it. At prediction time, a fresh record from the form goes through that exact same preparation before the trained model ever sees it.

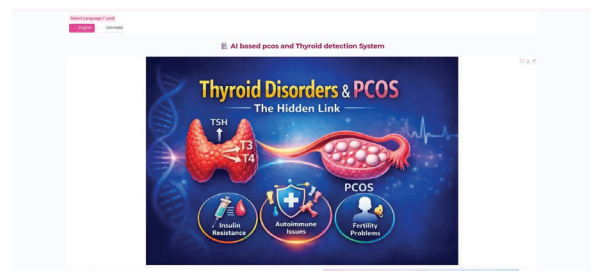


Fig. 1. Landing view of the deployed application — project title, language toggle, and the introductory PCOS–thyroid overview graphic.

We kept the modules fairly loosely coupled on purpose — the dataset, the preprocessing logic, the choice of classifier, and the interface layer can each be swapped or updated without having to rebuild everything else from scratch.

VII. DATASET AND FEATURE ENGINEERING

A. Dataset Organization

The dataset needs to contain women's health attributes alongside a labeled outcome for both PCOS and thyroid status. Depending on the exact data source used for the final training run, that outcome might be a simple binary label per condition or a combined multi-class target — either way, the actual record counts, class balance, and full feature list need to come from whichever dataset is finally used for training, not from the demo screenshots shown here.

B. Fields Collected Through the Interface

The deployed form itself gives a good sense of what the feature set looks like in practice. The Patient Profile section takes full name, age, district and taluk — the district/taluk fields are mainly there to help match a result to nearby specialists, rather than being fed into the classifier as predictive features. The PCOS section combines symptom checkboxes (irregular periods, severe acne, weight gain, thinning hair) with numeric fields for follicle count, AMH and HCG, plus a slider. The Thyroid section similarly combines symptom checkboxes (fatigue, cold sensitivity, neck swelling, rapid heartbeat) with TSH, T3 and TT4 readings.

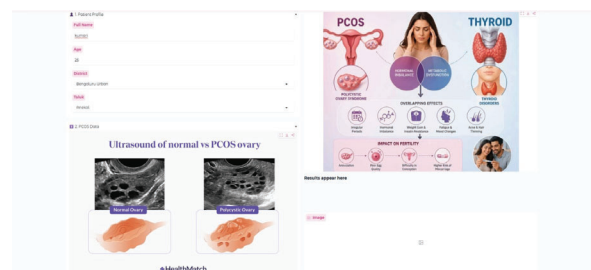


Fig. 2. Patient-profile entry (age, district, taluk) and the PCOS ultrasound comparison shown alongside the input form.

C. Feature Preparation

Checkbox-style symptoms are converted into binary indicator values, while the numeric hormone and thyroid readings are passed through largely as-is, subject to whatever scaling the final model needs. A field that isn't well supported by the dataset — or is missing too often to be reliable — gets dropped rather than just to look thorough.

VIII. DATA PREPROCESSING AND INPUT VALIDATION

Before training, the usual checks apply: duplicate rows, missing values, inconsistent categories and out-of-range numbers all get looked at, since a model trained on noisy data will happily learn the noise along with the signal. Any missing-value strategy — dropping records or imputing from the training partition — needs to be fixed and documented rather than decided ad hoc.

On the interface side, we also rely on basic client-side validation to keep obviously invalid values from ever reaching the model. Fig. 3 shows this in action — the T3 field rejects a value outside its accepted range and shows an inline hint telling the user the two nearest valid values, which is a small but useful safeguard against garbage input.

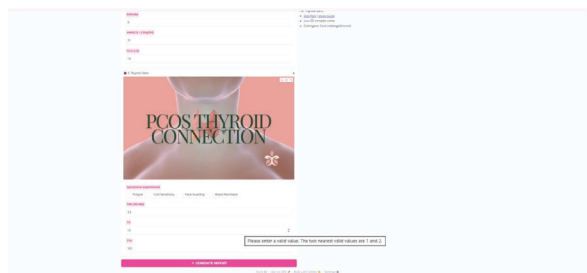


Fig. 3. Thyroid-data section with symptom checkboxes, TSH/T3/TT4 fields, and an inline validation message rejecting an out-of-range T3 entry.

IX. MACHINE LEARNING METHODOLOGY

A. Decision Tree

The Decision Tree works by repeatedly splitting the data on whichever feature best separates the classes at that point, starting from a root node and continuing down to leaf nodes that hold the final predicted class [5]. We're using it here mainly because its logic is easy to trace by hand — useful when the goal is to explain the system rather than just deploy it — though a tree grown too deep will start memorizing training examples instead of generalizing [4], so depth and leaf-size limits are worth tuning.

B. Random Forest

Random Forest builds many such trees, each on a randomly resampled subset of the data and features, and then combines their votes into one final answer [6]. Averaging over many trees this way tends to smooth out the quirks any single tree might pick up, which is generally a good fit for tabular data like ours [4].

C. Formal Representation

An input record is written as a feature vector $X = [x_1, x_2, \dots, x_n]$, with each x_i corresponding to one of the demographic, PCOS-related, or thyroid-related fields described earlier. Training learns a function $f(X) \rightarrow y$ mapping this vector to a target class y [7] — a sequence of split rules for the Decision Tree, or a majority vote across trees for the Random Forest. Any probability score the model reports should be read as an internal confidence estimate, not a clinically calibrated probability.

X. TRAINING, ALGORITHM, AND EVALUATION METRICS

Algorithm 1: End-to-end prediction procedure

1. Load the finalized dataset D and identify the target column(s).
2. Clean D and fix a preprocessing transformation T (encoding, scaling, missing-value handling).
3. Split D into training set D_{train} and test set D_{test} using a fixed random seed [8].
4. Fit T on D_{train} , transform D_{train} , and train classifier M (Decision Tree or Random Forest) on it.
5. Transform D_{test} with the same fitted T and evaluate M against the true labels [8].
6. On a new Gradio submission U , apply the same fitted T to U .
7. Compute $\hat{y} = M(U)$ and format it into the on-screen report.

Four standard classification metrics are used to judge the models once real test results are available [4], [7]:

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN)$$

$$\text{Precision} = TP / (TP + FP)$$

$$\text{Recall} = TP / (TP + FN)$$

$$F1 = 2 \times \text{Precision} \times \text{Recall} / (\text{Precision} + \text{Recall})$$

Here TP, TN, FP and FN are the true-positive, true-negative, false-positive and false-negative counts respectively. A confusion matrix rounds this out by showing exactly which classes get mixed up with which — something a single accuracy number can hide, especially if the dataset is imbalanced.

Stage	What happens in our pipeline
Loading	The chosen dataset is read into a structured data frame and its columns/target are inspected.
Cleaning	Duplicate rows, invalid entries, and missing fields are fixed or dropped based on a set policy.
Encoding	Category-type fields (symptom checkboxes, district/taluk, etc.) are converted to numeric form.
Feature–target split	Input attributes go into X while the label to be predicted becomes y .
Train–test split	A fixed random seed decides the training and testing partitions.
Model fitting	The Decision Tree and Random Forest classifiers are trained on the training portion.
Evaluation	Both models predict on the held-out test portion and metrics are computed.
Inference	The saved preprocessing and trained model are applied to a new Gradio form submission.

TABLE I. STAGES OF OUR DATA PREPROCESSING AND MODEL PIPELINE.

XI. SYSTEM IMPLEMENTATION

The whole prototype is written in Python, using Pandas and NumPy for data handling, scikit-learn [9] for the actual classifiers, Matplotlib for the contribution chart, and Gradio [3] for the front end — all run out of a Google Colab notebook during development. Gradio was a convenient choice here specifically because it let us get a shareable web link for the interface without setting up separate hosting.

As shown across Figs. 1–5, the layout keeps Patient Profile, PCOS Data and Thyroid Data as separate, collapsible sections, with a single Generate Report button that triggers the whole pipeline. We found this section-based layout easier for a first-time user to follow than one long scrolling form.

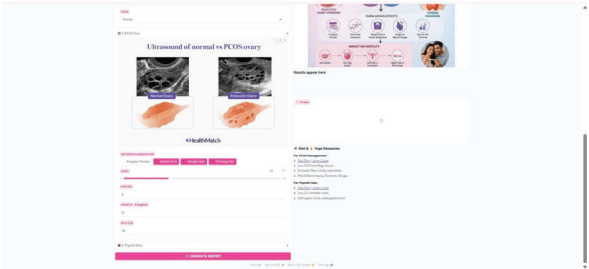


Fig. 4. PCOS section with symptoms selected (Severe Acne, Weight Gain, Thinning Hair), the follicle/AMH/HCG fields, and the diet & yoga resource panel.

XII. RESULTS AND DISCUSSION

We ran the deployed interface end to end with a demonstration record: a 25-year-old patient (“kumari”) from Anekal taluk in Bengaluru Urban district. On the PCOS side, the entered values were 6 follicles, an AMH of 2.1 ng/ml and an HCG of 1.9; on the thyroid side, TSH of 2.5, T3 of 1.5 and TT4 of 100, with Severe Acne, Weight Gain and Thinning Hair selected as symptoms. Submitting this through Generate Report returned PCOS: NEGATIVE and THYROID: NEGATIVE, alongside a list of nearby specialists and the Clinical Contribution Factors chart shown in Fig. 5.

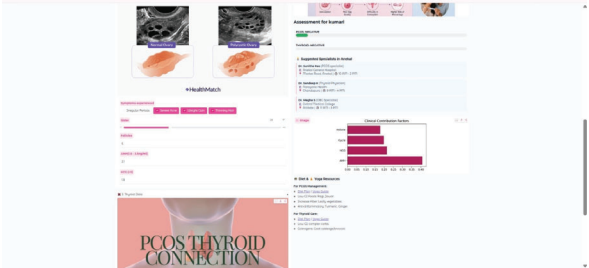


Fig. 5. Generated assessment for the demonstration case — PCOS: NEGATIVE, THYROID: NEGATIVE — with suggested local specialists and the Clinical Contribution Factors chart.

It's important to be honest about what this single run does and doesn't show. It confirms that the pipeline work end to end — the form accepts structured input, applies validation, runs it through the model, and returns a readable combined report. What it does not do is tell us anything about the model's general accuracy, since that requires testing on a proper held-out set with known labels [8], not one manually entered example. Table II summarizes the demonstration run, and Table III is where the real accuracy/precision/recall/F1 numbers belong once that evaluation has been completed — we've deliberately left it as placeholders rather than inventing numbers that aren't backed by an actual test run.

Item	Demonstrated value
Patient profile	Name: kumari; Age: 25; District: Bengaluru Urban; Taluk: Anekal
PCOS inputs	Symptoms: Severe Acne, Weight Gain, Thinning Hair; Follicles: 6; AMH: 2.1 ng/ml; HCG: 1.9
Thyroid inputs	TSH: 2.5; T3: 1.5; TT4: 100
Generated PCOS result	NEGATIVE
Generated thyroid result	NEGATIVE
Other report elements	Suggested local specialists, Clinical Contribution Factors chart, diet/yoga resources

TABLE II. SAMPLE DEMONSTRATION RUN (INPUTS AND GENERATED OUTPUT).

Model	Accuracy	Precision	Recall	F1-score
Decision Tree	[Final]	[Final]	[Final]	[Final]
Random Forest	[Final]	[Final]	[Final]	[Final]

TABLE III. FINAL MODEL PERFORMANCE — TO BE COMPLETED FROM THE ACTUAL TEST-SET EVALUATION.

A. Reading the Results Responsibly

Once real metrics are filled in, accuracy alone shouldn't decide which model is used — a model can look accurate overall while still missing a minority class if the dataset is imbalanced, so precision, recall, F1 and the confusion matrix all need to be looked at together. If Random Forest ends up with noticeably better recall while keeping precision reasonable, that would be a fair basis to prefer it over the Decision Tree — but only once those numbers actually exist.

Aspect	Decision Tree	Random Forest
Ease of reading	Very clear; one tree can be traced by eye.	Harder to trace directly since many trees are pooled.
Overfitting tendency	Rises quickly once the tree grows deep.	Kept in check by averaging many trees.
Fit for our tabular form data	Good.	Good.
Compute cost	Low.	Higher than a single tree.
Role in this project	Interpretable baseline.	Ensemble candidate for the deployed model.

TABLE IV. DECISION TREE VS. RANDOM FOREST — A QUALITATIVE COMPARISON.

XIII. SECURITY, PRIVACY, AND ETHICAL CONSIDERATIONS

Health data is sensitive by definition, and a prototype like this should collect no more personal information than it actually needs — the name/age/location fields here exist mainly to personalize the report and match specialists, not to build a profile of the user. If this is ever demonstrated publicly, we'd lean toward synthetic patient details rather than real ones, the way “kumari” is used as a placeholder name in our own testing.

The report screen is also careful, at least in intent, not to overstate what it's telling the user — it's a preliminary, algorithmic read, not a diagnosis, and it shouldn't be read as medication or treatment advice. Anyone using it with real symptoms or concerns should still be pointed toward an actual doctor, which is exactly what the specialist-suggestion feature

is meant to nudge toward.

XIV. LIMITATIONS

- Everything the model predicts is only as good as the dataset it was trained on — a small or unrepresentative dataset will show up as unreliable predictions later.
- A single demonstration run, like the one shown in Section XII, cannot establish how reliable the system is in general.
- Self-reported symptoms (checkboxes) carry the usual risk of being inaccurate or inconsistently interpreted by different users.
- The Clinical Contribution Factors chart and the diet/yoga suggestions are convenience features and haven't been clinically validated.
- Results from one dataset or one region's patient population may not carry over well elsewhere.
- The prototype has not gone through any process that would let it be called a validated diagnostic tool.

XV. FUTURE ENHANCEMENTS

- Train and test on a larger, more diverse dataset gathered with proper ethical approval.
- Run a full evaluation with cross-validation [8] and report real accuracy/precision/recall/F1 numbers.
- Try additional models — Logistic Regression, SVM, Gradient Boosting — for comparison against the two tree-based methods.
- Add explainable-AI output so a user can see why a particular prediction was made, not just the chart of contributing fields.
- Validate the specialist-matching and diet/yoga content with actual clinical or nutrition input.
- Harden the deployment with authentication and proper storage before handling any real patient data.

XVI. CONCLUSION

This project put together a working, end-to-end screening tool for PCOS and thyroid disorders — from data preprocessing through Decision Tree and Random Forest classification to a Gradio interface that a non-technical user can actually operate. The screenshots and sample run included here show that the pipeline functions as intended: a patient's details go in, get validated and transformed, and come back as a readable report with a prediction, specialist suggestions and lifestyle pointers.

At the same time, we've tried to be careful not to oversell what a single demonstration case can prove. The PCOS: NEGATIVE / THYROID: NEGATIVE result shown here is evidence that the system runs correctly, not evidence of how accurate it is — that still needs a proper held-out test set and the metrics discussed in Section X. With that evaluation done, and with the limitations in Section XIV kept in mind, we think this prototype is a reasonable foundation to build further validation and, eventually, real clinical input on top of.

ACKNOWLEDGMENT

We would like to thank Prof. Mamatha L, Assistant Professor, Department of Computer Science and Engineering, Shridevi Institute of Engineering and Technology, Tumakuru, for her guidance and patience through the many revisions of this project. We're also grateful to the Department of CSE and the institution for the environment and resources that made this

work possible.

REFERENCES

- [1] World Health Organization, "Polycystic ovary syndrome," WHO Fact Sheet. [Online]. Available: <https://www.who.int/news-room/fact-sheets/detail/polycystic-ovary-syndrome>. Accessed: Sep. 29, 2026.
- [2] National Institute of Diabetes and Digestive and Kidney Diseases, "Thyroid disease," NIDDK Health Information. [Online]. Available: <https://www.niddk.nih.gov>. Accessed: Sep. 29, 2026.
- [3] Gradio Team, "Gradio: Build machine learning web apps in Python," Gradio Documentation. [Online]. Available: <https://www.gradio.app/docs>. Accessed: Sep. 29, 2026.
- [4] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York, NY, USA: Springer, 2009.
- [5] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.
- [6] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.
- [7] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.
- [8] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proc. 14th Int. Joint Conf. Artificial Intelligence (IJCAI)*, 1995.
- [9] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

APPENDIX A — FINAL EXPERIMENTAL INFORMATION TO INSERT

Before this paper is submitted for final review, replace the placeholders below with values from the completed training run:

- Final dataset name, source, and total record count.
- Class distribution for PCOS and thyroid labels.
- Exact list of input features used in the final model.
- Missing-value and encoding strategy actually applied.
- Train-test split ratio and random seed.
- Decision Tree and Random Forest hyperparameters.
- Final accuracy, precision, recall, F1-score and confusion matrix for both models.

APPENDIX B — RESPONSIBLE USE STATEMENT

This project is an academic prototype. Its predictions come from statistical patterns learned from a training dataset and are not confirmed medical diagnoses. Anyone with real symptoms or concerns about PCOS or thyroid disorders should consult a qualified healthcare professional. No claim of clinical effectiveness or diagnostic accuracy is made unless that validation is separately carried out and documented.

AUTHOR INFORMATION

Amrutha D R – 1SV23CS004
Kavyashree M – 1SV23CS040
Kushala S – 1SV23CS044
Meghanasmitha L – 1SV23CS060
Department of Computer Science and Engineering, Shridevi Institute of Engineering and Technology, Tumakuru, Karnataka, India